ECE 407
Computer Aided Design for Electronic Systems

Introduction
Instructor: Maria K. Michael

Overview

• Introduction to Electronic Systems, their requirements and trends
• Design Styles of ASICs
  – Full Custom
  – Standard Cell
  – Gate Arrays
  – Programmable Logic Devices
  – Field-Programmable Gate Arrays
• Computer-Aided Design
  – Design Flow
  – Abstraction Models
• ASIC Cell Libraries
Electronic Systems

- **Embedded Systems / Systems on Chip** are everywhere
- Technology advances enable the design of more complex systems
- Challenges:
  - Catch-up with technology
  - Cope with complexity

Application of Microcontrollers

<table>
<thead>
<tr>
<th>Year</th>
<th>Home</th>
<th>Office</th>
<th>Automobile</th>
</tr>
</thead>
<tbody>
<tr>
<td>1970</td>
<td>2</td>
<td>18</td>
<td>2</td>
</tr>
<tr>
<td>1980</td>
<td>9</td>
<td>42</td>
<td>5</td>
</tr>
<tr>
<td>1990</td>
<td>59</td>
<td>35</td>
<td>10</td>
</tr>
<tr>
<td>2000</td>
<td>226</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

- **Home:** Appliances, Intercom, Telephones, Security systems, Garage door opener, Answering machines, Fax machines, Home computers, TVs, VCRs, Camcorder, Remote Control, Video games, Cellular phones, Musical Instruments, Sewing machines, Lighting control, Paging, Cameras, Exercise Equipment
- **Office:** Telephones, Computers, Security System, Fax machines, Copiers, Printers, Remote control, Pagers
- **Automobile:** Air bags, ABS, Security System, Drive by wire, Trap computer, Instrumentation, Transmission control, Entertainment, Climate control, Keyless entry, GPS

- Average # of microcontroller ICs
Gordon Moore’s Law

Moore’s Law (1965):
“The number of transistors per square inch will double every 18 months”

The Number of Transistors Per Chip Double Every 18 Months

Gordon Moore’s Law (con’t …)

Microprocessor Transistor Counts 1971-2011 & Moore’s Law
Gordon Moore’s Law

Moore’s Law (1965): “The number of transistors per square inch will double every 18 months.”

Growth of capacity per DRAM chip

Year of Introduction

Integration Density: Moore’s Law

Why is memory capacity larger than CPU capacity?
Microprocessor Performance Trends

What is causing the frequency increase? It's two-fold ...

Technology Progress: Another way to look at it...

Moore's Law: Capacity doubles every 18 months

Price of 1 Mbit memory

<table>
<thead>
<tr>
<th>Year</th>
<th>Price (Euro)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1973</td>
<td>75,000</td>
</tr>
<tr>
<td>1977</td>
<td>5,000</td>
</tr>
<tr>
<td>1981</td>
<td>400</td>
</tr>
<tr>
<td>1994</td>
<td>120</td>
</tr>
<tr>
<td>1997</td>
<td>30</td>
</tr>
<tr>
<td>1990</td>
<td>6</td>
</tr>
<tr>
<td>1996</td>
<td>0.6</td>
</tr>
<tr>
<td>2000</td>
<td>0.06</td>
</tr>
</tbody>
</table>
Processors: Then and Now …

A “closer” look …
Integrated Systems

- **Silicon Technology (CMOS)**
  - Downscaling of feature sizes
  - Nanotechnologies already here...

- **Different Design Styles**
  - To address performance and cost issues

---

Integrated Systems

- **Systems on Chip (SOC)**
  - Multi-processing SoCs (MPSoCs)
  - Chip Multi-Processors (CMPs)

- **Systems in a Package (SiP)**

- **3D Chips**
Application Specific Integrated Circuits (ASICs)

- IC – made on a thin circular silicon called wafer, that holds hundreds of die.
- Circuits are build up with successive mask layers (usually 10-15 layers, half for transistors and half for interconnect)

A gate equivalent is a 2-input NAND gate (F = (a•b)', 4 transistors in CMOS)
- IC size determined by the # of gates (equivalent) on the chip
- History of Integration:
  - SSI (~ 10 gates)
  - MSI (~ 100-1,000 gates)
  - LSI (~ 1,000-10,000 gates)
  - VLSI (~ 10,000-100,000 gates)
  - ULSI (~ 1M-xB gates)
- History of Technology:
  - Bipolar and TTL preceded CMOS technology due to the difficulty in making metal-gate n-channel MOS (nMOS); introduction of CMOS reduced power greatly!
- Feature size (λ): smallest shape you fabricate on a chip (roughly, half the length of the smallest transistor for 0.5µm process, λ = 0.25 µm)
Design Styles of ASICs

Complexity of VLSI circuits

- Performance
- Size
- Cost
- Market time

Different design styles

- Full custom
- Gate array
- Standard cell
- FPGA

Cost
Flexibility

Semi-custom

Full-Custom ASICs

Intel 4004
Full-Custom ASICs

Intel 4004 ('71)

Intel 8080

Intel 8085

Intel 8286

Intel 8486

Full-Custom ASICs …

4th Generation Intel® Core™ Processor Die Map
22nm Haswell Tri-Gate 3-D Transistors

Processor Graphics

Core

Core

Core

Core

Shared L3 Cache**

Quad core die shown above

Transistor count: 1.4Billion

Die size: 177mm²

** Cache is shared across all 4 cores and processor graphics.
Full-Custom ASICs

- Designers do not use pre-defined/pre-tested cells
- All mask layers are customized
- **Advantages:**
  - Highest performance
  - Lowest *part* cost
  \[\text{Optimized w.r.t.}\]
  \[\text{delay, area, power}\]
- **Disadvantages:**
  - Increased design time, design expense
  - Increased complexity, highest risk
- Microprocessors used to be exclusively full-custom; designers are increasingly turning to semi-custom ASIC design styles

Standard Cell-Based ASICs

Routing channel requirements are reduced by presence of more interconnect layers
Standard Cell-Based ASICs

- Pre-designed logic cells (e.g., AND, OR, MUX, FF), known as **Standard Cells (SC)**
- Standard cell area is built of **rows** of SCs (=flexible blocks)
- Can have more than one SC area and can decide placement
- SC area may be used in combination with large **pre-designed** cells called **megacells** (or **fixed blocks**)
- All mask layers are customized (allows for transistor re-sizing)
- Custom blocks can be embedded
- Advantage:
  - Save time using the SC library
    (cells are pre-designed, pre-tested, pre-characterized)

Gate-Array ASICs

![Gate Array Diagram](image-url)
Gate-Array ASICs

- Transistors are pre-defined on the silicon wafer
- Pre-defined pattern of transistors forms the base array
- **Base cell** = smallest element replicated to make the base array
- Designers use custom masks to define ONLY the few top metal layers (interconnect)
- → logic is pre-fabricated, only interconnect must be defined and fabricated
- **Macro** = logic cells in a gate array library

3 types of Gate-Arrays (GAs)

- **Channeled GA**
  - space between BA rows for wiring (routing)
  - channel width is fixed (variable in standard-cells)
- **Channeless GA**
  - no channels for wiring
  - uses unused transistors or over-the-cell for routing interconnect
  - logic density (= # of logic-on-silicon) is higher than in channeled GAs
- **Structured GA**
  - can embed a custom block (eg. RAM), fixed size
  - GA area can be channeled or chaneless
Programmable Logic Devices (PLDs)

- Standard ICs available in standard configuration.
- Still considered ASICs, since they are programmed to create customized parts.
- No need to customize mask layers or logic cells.
- Fast design turn-around.
- 1 single block of programmable interconnect.
- **Macrocell** = programmable logic array (AND/OR array) followed by FFs/latches.
- Different technologies for programming (permanent or re-programmable).

PROM w/ 8 words x 4 bits

PAL w/ 3 Inputs, 4 Outputs and 8 Products
Programmable Logic Devices (PLDs)

- PLA w/ 3 Inputs, 4 Outputs and 8 Products
- PLS w/ 2 Inputs and 3 Outputs

Field Programmable Gate Arrays (FPGAs)

- In terms of functionality:
  - a large PLD! (also known as CPLD)
- Major difference with PLDs: programmable interconnect is NOT a single block
- No mask customization
- Both logic and interconnect can be programmed
- I/O cells surrounding the cells can also be programmed
- To learn more: see ECE 408: Digital Design w/ FPGAs
Field Programmable Gate Arrays (FPGAs)

Fuse-based Xilinx 4000 Interconnect Architecture
Comparison of Design Styles

<table>
<thead>
<tr>
<th>style</th>
<th>full-custom</th>
<th>standard cell</th>
<th>gate array</th>
<th>FPGA</th>
</tr>
</thead>
<tbody>
<tr>
<td>cell size</td>
<td>variable</td>
<td>fixed height*</td>
<td>fixed</td>
<td>fixed</td>
</tr>
<tr>
<td>cell type</td>
<td>variable</td>
<td>variable</td>
<td>fixed</td>
<td>programmable</td>
</tr>
<tr>
<td>cell placement</td>
<td>variable</td>
<td>in row</td>
<td>fixed</td>
<td>fixed</td>
</tr>
<tr>
<td>interconnections</td>
<td>variable</td>
<td>variable</td>
<td>variable</td>
<td>programmable</td>
</tr>
</tbody>
</table>

*uneven height cells are also used.

Computer-Aided Design

- Enabling design methodology
  - Support large scale system design
  - Design optimization
  - Reduced design time and time to market

- “… the only purpose of science is to ease the hardship of human existence…”
  (in Gallileo by Brecht)
### VLSI Realization Process

**Customer Needs**

- **Determine Requirements**
- **Write Specifications**
- **Design & Test Development**
- **Fabrication**
- **Manufacturing Test**

**Needs to be satisfied by the chip, i.e., function of the application**

**Determine Chip characteristics:**
- Function (I/O),
- Operating (power, frequency, noise, etc),
- Physical (packaging, etc),
- Environmental (temperature, reliability, etc),
- Other (volume, cost, price, availability, etc).

**Good Chips to Customer**
VLSI Realization Process

Customer Needs

- Determine Requirements
- Write Specifications
- Design & Test Development
- Fabrication
- Manufacturing Test

Architectural Design/Test
- System/Macro-level model
  - Verification
  - Logic/Gate-level model
  - Physical Design/Test
  - Transistor/Device-level model

Apply Test, detect/locate fabrication defects

Good Chips to Customer

Hierarchy of design abstractions for ICs

Abstraction Model:
- English
- executable program
- sequential machines
- logic gates
- transistors
- rectangles

Measuring Unit:
- function
- register-transfer
- logic
- circuit
- layout

- System throughput, design time
- Function units, clock cycles
- Literals, gate depth, power
- nanoseconds
- microns

cost
Design Flow

1. enter design using HDL or schematic
2. use HDL/schematic to produce architectural-level description
3. produce netlist (logic-level description)
4. verify/validate correctness of design (logic verification)
5. develop test strategy (includes fault modeling, test generation, design-for-testability)
6. divide large system into blocks of netlists
7. floorplan chip area – arrange blocks
8. decide exact location of cells in a block
9. connect cells and blocks (global and detailed routing)
10. determine resistance/capacitance of interconnect
11. verify design with the added loads of interconnect
ASIC Cell Libraries

- 3 choices (for gate-arrays & standard-logic designs)
- Use a design kit from the ASIC vendor (cheap)
  - Phantom library = cells are empty, enough info for layout; netlist is returned to vendor to fill in the boxes (initiation)
  - Must use tools approved by the vendor
- Buy an ASIC-vendor library (expensive)
  - Requires a qualified cell library
  - If you own the masks you have a customer owned tooling solution
- Build the cell library (very expensive)
  - Very time consuming
  - Requires: cell layout, behavioral model for simulation, HDL model, timing model, test strategy, cell schematic, cell icon/symbol, wire-load model, routing model, per cell
- For FPGAs, vendor supplies the cell library (designer kit)